Amendments to the Claims:

The following listing of claims will replace all prior versions, and listings, of claims in the application:

1. (Currently Amended) A computer-implemented method of operating a computer system for evaluating a programming language statement to solve a computational related problem relating to a data structure, the programming language statement including~~that includes~~ first and second sub-statements that provide a common framework to define both imperative and declarative statements providing an intermediate level of abstraction that enables concise declarative notation while preserving low level imperative specifications through bi-valuation, the method comprising ~~the steps of~~:

inputting a computer-readable arbitrary complex data structure representing an expression;

inputting the programming language statement, including the first and second sub-statements, that acts on the data structure to solve a computational related problem, one of the first and second sub-statements being a declarative statement and the other being an imperative statement;

evaluating the first sub-statement as a first evaluation to determine a first evaluation success result if the first evaluation succeeds or else a distinguished value if the first evaluation ~~fails; said~~fails, said distinguished value being a value not included in a first range of possible evaluation success results of the first sub-statement;

determining whether to evaluate the second sub-statement as a second evaluation, and if so, evaluating the second sub-statement to determine a second evaluation success result if the second evaluation succeeds or else said distinguished value if the second evaluation ~~fails; a~~fails, a second range of possible evaluation success results of the second sub-statement not including said distinguished value; ~~and~~

determining an evaluation result of the statement depending on at least whether the first evaluation of the first sub-statement succeeds or fails;

wherein the first evaluation success result of the first sub-statement and the second evaluation success result of the second sub-statement belong to a first type in a set of types and the distinguished value belongs to a second type not in the set of types,

wherein each type in the set of types is not a supertype of the second type; and

wherein the first type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and ~~enumeration.~~enumeration; and

outputting the evaluation results of the programming language statement to solve the computational related problem relating to the expression expressed in the arbitrary complex data structure.

2. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated if the first evaluation of the first sub-statement did not fail, and the evaluation result of the statement is determined to be the second evaluation success result of the second sub-statement if the first and second evaluations of the respective first and the second sub-statements succeeds, and wherein the evaluation result of the statement is said distinguished value if at least one of the first and second evaluations of the respective first and second sub-statements fails.

3. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated if the first evaluation of the first sub-statement fails, and wherein the evaluation result of the statement is the first evaluation success result of the first sub-statement if the first evaluation of the first sub-statement succeeds; the evaluation result of the statement is the second evaluation success result of the second sub-statement if the first evaluation of the first sub-statement fails but the second evaluation of the second sub-statement succeeds; and the evaluation result of the statement is said distinguished value if both of the first and second evaluations of the respective first and second sub-statements fail.

4. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated concurrently with the first evaluation of the first sub-statement, and the evaluation result of the statement is said distinguished value if at least one of the first and second evaluations of the respective first and second sub-statements fails.

5. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated concurrently with the first evaluation of the first sub-statement, and the evaluation result of the statement is said distinguished value only if both of the first and second evaluations of the respective first and second sub-statements fail.

6. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated independently on whether the first evaluation of the first sub-statement succeeds, and the evaluation result of the statement is said distinguished value if at least one of the first and second evaluations of the respective first and second sub-statements fails.

7. (Previously Presented) The method of claim 1, wherein the second sub-statement is evaluated independently on whether the first evaluation of the first sub-statement succeeds, and the evaluation result of the statement is said distinguished value if both of the first and second evaluations of the respective first and second sub-statements fail.

8. (Previously Presented) The method of claim 1, wherein at least one of the first and second sub-statements includes a closure loop statement having an operand indicating that evaluation of the respective sub-statement continues at least until said operand evaluates to said distinguished value.

9. (Previously Presented) The method of claim 1, wherein at least one of the first or second sub-statements includes a rule statement having a first argument and a second argument, a first argument evaluation of the first argument triggering a second argument evaluation of the second argument.

10. (Original) The method of claim 1, wherein at least one of the first or second sub-statements includes an ordered action system.

11. (Original) The method of claim 1, wherein at least one of the first or second sub-statements includes an unordered action system.

12. (Canceled)

13. (Original) The method of claim 1, wherein the first and second sub-statements are typed according to a hierarchy of types.

14. (Original) The method of claim 13, wherein said hierarchy of types includes at least one minimal type.

15. (Currently Amended) An article of manufacture for use in a computer system to provide an intermediate level of abstraction that enables concise declarative notation while preserving low level imperative specifications through bi-valuation, comprising:

a ~~computer readable~~ computer-readable storage device;~~device on a tangible medium;~~

instructions stored in the computer readable storage device for operating a method for evaluating a programming language statement that includes a first and a second sub-statement that provide a common framework to define both imperative and declarative statements, the method comprising:

inputting a computer-readable arbitrary complex data structure representing an expression;

inputting the programming language statement, including the first and second sub-statements, that acts on the data structure to solve a computational related problem, one of the first and second sub-statements being a declarative statement and the other sub-statement being an imperative statement;

evaluating the first sub-statement as a first evaluation to determine a first evaluation success result if the first evaluation succeeds or else a distinguished value if the first evaluation ~~fails; said~~fails, said distinguished value being a value not included in a first range of possible evaluation success results of the first sub-statement;

determining whether to evaluate the second sub-statement as a second evaluation, and if so, evaluating the second sub-statement to determine a second evaluation success result if evaluation succeeds or else said distinguished value if the second evaluation ~~fails; a~~fails, a second range of possible evaluation success results of the second sub-statement not including said distinguished value; ~~and~~

determining an evaluation result of the statement depending on at least whether the first evaluation of the first sub-statement succeeds or fails;

wherein the first evaluation success result of the first sub-statement and the second evaluation success result of the second sub-statement belong to a first type in a set of types and the distinguished value belongs to a second type not in the set of types,

wherein the first type in the set of types is not a supertype of the second type; and

wherein each type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and ~~enumeration.~~enumeration; and

outputting the evaluation results of the programming language statement to solve the computational related problem relating to the expression expressed in the arbitrary complex data structure.

16. (Currently Amended) A computer ~~executable~~ system for evaluating a programming language statement and determining an evaluation result of said statement to ~~provide~~ solve a computational related problem related to an arbitrary complex data structure representing an expression by providing an intermediate level of abstraction that enables concise declarative notation while preserving low level imperative specifications through bi-valuation; comprising:

a computer readable storage device configured to store an arbitrary complex data structure representing an expression and ~~the~~a programming language statement that includes a first and a second sub-statement that provide a common framework to define both imperative and declarative statements, one of the first and second sub-statements being a declarative statement and the other being an imperative statement;

a processor that receives the computer-readable arbitrary complex data structure and determines the evaluation result of the ~~statement~~programming language statement that acts on the arbitrary complex data structure; the evaluation result ~~of the statement~~ depending on whether a sub-evaluation of the first and second sub-statements succeeds or fails; the processor evaluating the first sub-statement and determining a first evaluation success result if the sub-evaluation succeeds, or a distinguished value if evaluation fails; the processor evaluating the second sub-statement and determining a second evaluation success result if the sub-evaluation succeeds, or said distinguished value if the sub-evaluation fails; the first evaluation success result of the first sub-statement and the second evaluation success result of the second sub-statement belong to a first type in a set of types and the distinguished value belongs to a second type not in the set of types; wherein each type in the set of types is not a supertype of the second type; ~~and~~ wherein the first type in the set of types comprises one from the set of integer, float, Boolean, sequence, tuple, structure, multi-set, dictionary, string and ~~enumeration.~~enumeration; and the processor outputting the evaluation results of the programming language statement evaluations to solve the computational related problem related to the expression expressed in the arbitrary complex data structure.

17-18. (Cancelled)

19. (Previously Presented) The method of claim 1, wherein evaluation of at least one of the first sub-statement and the second sub-statement comprises a pattern matching operation.